Improving Zero-shot Generalization in Offline Reinforcement Learning using Generalized Similarity Functions

Bogdan Mazoure[*]
McGill University and Quebec AI Institute
bogdan.mazoure@mail.mcgill.ca

Ilya Kostrikov
UC Berkeley

Ofir Nachum
Google Brain

Jonathan Tompson
Google Brain

Abstract

Reinforcement learning (RL) agents are widely used for solving complex sequential decision making tasks, but still exhibit difficulty in generalizing to scenarios not seen during training. While prior online approaches demonstrated that using additional signals beyond the reward function can lead to better generalization capabilities in RL agents, i.e. using self-supervised learning (SSL), they struggle in the offline RL setting, i.e. learning from a static dataset. We show that performance of online algorithms for generalization in RL can be hindered in the offline setting due to poor estimation of similarity between observations. We propose a new theoretically-motivated framework called Generalized Similarity Functions (GSF), which uses contrastive learning to train an offline RL agent to aggregate observations based on the similarity of their expected future behavior, where we quantify this similarity using *generalized value functions*. We show that GSF is general enough to recover existing SSL objectives while also improving zero-shot generalization performance on a complex offline RL benchmark, offline Procgen.

1 Introduction

Reinforcement learning (RL) is a powerful framework for solving complex tasks that require a sequence of decisions. The RL paradigm has allowed for major breakthroughs in various fields, e.g. outperforming humans on video games [Mnih et al., 2015, Schwarzer et al., 2020], controlling stratospheric balloons [Bellemare et al., 2020] and learning reward functions from robot manipulation videos [Chen et al., 2021]. More recently, RL agents have been tested in a generalization setting, i.e. in which training involves a finite number of related tasks sampled from some distribution, with a potentially distinct sampling distribution during test time [Cobbe et al., 2019, Song et al., 2019]. The main issue for designing generalizable agents is the lack of on-policy data from tasks not seen during training: it is impossible to enumerate all variations of a real-world environment during training and hence the agent must extrapolate from a (limited) training task collection onto a broader set of problems. Since the learning agent is given no training data from test-time tasks, this problem is referred to as zero-shot generalization. In our work, we are interested in the problem of zero-shot generalization where the difference between tasks is predominantly due to perceptually distinct observations. An example of this setting is any environment with distractor features [Cobbe et al., 2020, Stone et al., 2021], i.e. features with no dependence on the reward signal nor the agent's decisions. This generalization setting has recently received much attention, due to its particular rel-

[*]Work done while at Google Brain.

Offline Reinforcement Learning Workshop at Neural Information Processing Systems, 2021.

evance to real-world scenarios, for example deploying a single autonomous driving agent at day or at night [Liu et al., 2020a, Agarwal et al., 2021, Mazoure et al., 2021a].

Generalization capabilities of an agent can be analyzed through the prism of *representation learning*, under which the agent's current belief about a rich and high-dimensional environment are summarized in a low-dimensional entity, called a representation. Recent work in online RL has shown that learning state representations with specific properties such as disentanglement [Higgins et al., 2017] or linear separability [Lin et al., 2020] can improve zero-shot generalization performance. Achieving this with limited data (i.e. offline RL) is challenging, since the representation will have a large estimation error over regions of low data coverage. A common solution to mitigate this task-specific overfitting and extracting the most information out of the data consists in introducing auxiliary learning signals other than instantaneous reward [Raileanu and Fergus, 2021]. As we show later in the paper, many such signals already contained in the dataset can be used to further improve generalization performance. For instance, the generalization performance of PPO on Procgen remains limited even when training on 200M frames, while generalization-oriented agents [Raileanu and Fergus, 2021, Mazoure et al., 2021a] can outperform it by leveraging additional auxiliary signals. However, a major issue with the aforementioned methods is their exorbitant reliance on online access to the environment, an impractical restriction for real-world scenarios.

In contrast, in many real-world scenarios access to the environment is restricted to an offline, fixed dataset of experience [Ernst et al., 2005, Lange et al., 2012]. A natural limitation for generalization from offline data is that policy improvement is dependent on dataset quality. Specifically, high-dimensional problems such as control from pixels require large amounts of training experience: a standard training of PPO [Schulman et al., 2017] for 25 million frames on Procgen [Cobbe et al., 2020] generates more than 300 Gb of data, an impractical amount of data to share for offline RL research. Improving zero-shot generalization performance from an offline dataset of high-dimensional observations is therefore a hard problem due to limitations on dataset size and quality.

In this work, we are interested in improving zero-shot generalization across a family of Partially-Observable Markov decision processes [POMDPs, Murphy, 2000] in an offline RL setting, i.e. by training agents on a fixed dataset. We hypothesize that in order for an RL agent to be able to generalize across perceptually different POMDPs without adaptation, observations with similar future behavior should be assigned to close representations. We use the generalized value function (GVF) framework [Sutton et al., 2011] to capture future behavior with respect to an arbitrary instantaneous signal (called cumulant) for a given state. The choice of cumulant then determines the nature of the behavioral similarity that is encouraged for generalization. For example, using reward as the signal gives rise to of reward-aware behavioral similarity such as bisimulation [Ferns et al., 2004, Li et al., 2006, Castro, 2020, Zhang et al., 2020]; using future state-action counts encourages reward-free behavioral similarity [Misra et al., 2020, Liu et al., 2020a, Agarwal et al., 2021, Mazoure et al., 2021a].

Our main contributions are as follows:

1. We propose Generalized Similarity Functions (GSF), a novel self-supervised learning algorithm for reinforcement learning, that aggregates latent representations by the future behavior (or generalized value function) under their respective observations.
2. We devise a new benchmark constructed to test zero-shot generalization of offline RL algorithms: offline Procgen. It consists of 5M transitions from 200 related levels of 16 distinct games sampled from the "easy" game mode.
3. We evaluate performance of GSF and other baseline methods on offline Procgen, and show that GSF outperforms both previous state-of-the-art offline RL and representation learning baselines on the entire distribution of levels.
4. We analyze the theoretical properties of GSF and describe the impact of hyperparameters and cumulant functions on empirical behavior.

2 Related Works

Generalization in reinforcement learning Generalizing a model's predictions across a variety of unseen, high-dimensional inputs has been extensively studied in the static supervised learning setting [Bartlett, 1998, Triantafillou et al., 2019, Valle-Pérez and Louis, 2020, Liu et al., 2020b]. Gener-

alization in RL has received a lot of attention: extrapolation to unseen rewards [Barreto et al., 2016, Misra et al., 2020], observations [Zhang et al., 2020, Raileanu and Fergus, 2021, Liu et al., 2020a, Agarwal et al., 2021, Mazoure et al., 2021a] and transition dynamics [Ball et al., 2021]. Each generalization scenario is best solved by their respective set of methods: sufficient exploration [Misra et al., 2020, Agarwal et al., 2020], auxiliary learning signals [Srinivas et al., 2020, Mazoure et al., 2020, Stooke et al., 2021] or data augmentation [Ball et al., 2021, Sinha and Garg, 2021]. Data augmentation is a promising technique, but typically relies on handcrafted domain information, which might not be available *a priori*. In fact, we will show in our experiments that generalization in the offline RL setting is poor even when using such handcrafted data augmentations, without additional representation learning mechanisms. In this work, we posit that representation learning should use instantaneous auxiliary signals in order to prevent overfitting onto a unique signal (e.g. reward across tasks) and improve generalization performance. Theoretical generalization guarantees have only been provided so far for limited scenarios, mostly for bandits [Swaminathan and Joachims, 2015], linear MDPs [Boyan and Moore, 1995, Wang et al., 2021b, Nachum and Yang, 2021] and across reward functions [Castro and Precup, 2010, Barreto et al., 2016, Wang et al., 2021a, Touati and Ollivier, 2021].

Representation learning For simple POMDPs, near-optimal policies can be found by optimizing for the reward alone. However, more complex settings may require additional auxiliary signals in order to find state abstractions better suited for control. The problem of learning meaningful state representations (or abstractions) for planning and control has been extensively studied previously [Jong and Stone, 2005, Li et al., 2006], but saw real breakthroughs only recently, in particular due to advances in self-supervised learning (SSL). Outside of RL, SSL has achieved spectacular results by closing the gap between unsupervised and supervised learning on certain tasks [Hjelm et al., 2018, Oord et al., 2018, Caron et al., 2020, Grill et al., 2020]. Self-supervised representation learning has also been used to achieve state-of-the-art generalization and sample efficiency results in RL on challenging control problems such as data efficient Atari [Schwarzer et al., 2020, 2021], DeepMind Control [Agarwal et al., 2021] and Procgen [Mazoure et al., 2020, Stooke et al., 2021, Raileanu and Fergus, 2021, Mazoure et al., 2021a]. Noteworthy instances of theoretically-motivated representation learning methods for RL include heuristic-guided learning [Sun et al., 2018, Mazoure et al., 2021b, Cheng et al., 2021], and random Fourier features [Nachum and Yang, 2021].

Offline reinforcement learning When learning from a static dataset, agents should balance interpolation and extrapolation errors, while ensuring proper diversity of actions (i.e. prevent collapse to most frequent action in the data). Popular offline RL algorithms such as BCQ [Fujimoto et al., 2019], MBS [Liu et al., 2020c], and CQL [Kumar et al., 2020] rely on a behavior regularization loss [Wu et al., 2019] as a tool to control the extrapolation error. Some methods, such as F-BRC [Kostrikov et al., 2021] are defined only for continuous action spaces while others, such as MOReL [Kidambi et al., 2020] estimate a pessimistic transition model. The major issue with current offline RL algorithms such as CQL is that they are perhaps overly pessimistic for generalization purposes, i.e. CQL and MBS ensure that the policy improvement is well-supported by the batch of data. As we will show in our empirical comparisons, using overly conservative policy updates can prevent the representation from fully leveraging the information of the training dataset.

3 Problem setting

3.1 Partially-observable Markov decision processes

A (infinite-horizon) partially-observable Markov decision process [POMDP, Murphy, 2000] M is defined by the tuple $M = \langle \mathcal{S}, p_0, \mathcal{A}, p_{\mathcal{S}}, \mathcal{O}, p_{\mathcal{O}}, r, \gamma \rangle$, where $\mathcal{S}$ is a state space, $p_0 = \mathbb{P}[s_0]$ is the starting state distribution, $\mathcal{A}$ is an action space, $p_{\mathcal{S}} = \mathbb{P}[\cdot|s_t, a_t] : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is a transition function, $\mathcal{O}$ is an observation space, $p_{\mathcal{O}} = \mathbb{P}[\cdot|s_t] : \mathcal{S} \to \Delta(\mathcal{O})$[2] is an observation function, $r : \mathcal{S} \times \mathcal{A} \to [r_{\text{min}}, r_{\text{max}}]$ is a reward function and $\gamma \in [0, 1)$ is a discount factor. The system starts in one of the initial states $s_0 \sim p_0$ with observation $o_0 \sim p_{\mathcal{O}}(\cdot|s_0)$. At every timestep $t = 1, 2, 3, ..,$ the agent, parameterized by a policy $\pi : \mathcal{O} \to \Delta(\mathcal{A})$, samples an action $a_t \sim \pi(\cdot|o_t)$. The environment transitions into a next state $s_{t+1} \sim p_{\mathcal{S}}(\cdot|s_t, a_t)$ and emits a reward $r_t = r(s_t, a_t)$ along with a next observation $o_{t+1} \sim p_{\mathcal{O}}(\cdot|s_{t+1})$.

[2] $\Delta(\mathcal{X})$ denotes the entire set of distributions over the space $\mathcal{X}$.

The goal of an RL agent is to maximize the cumulative rewards $\sum_{t=0}^{\infty} \gamma^t r_t$ obtained over the entire episode. Value-based off-policy RL algorithms achieve this by estimating the state-action value function under a target policy π:

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{\mathbb{P}_t^{\pi}}[\sum_{k=1}^{\infty} \gamma^k r(s_{t+k}, a_{t+k}) | s_t, a_t], \ \forall s_t \in \mathcal{S}, a_t \in \mathcal{A} \tag{1}$$

where $\mathbb{P}_t^{\pi}$ denotes the joint distribution of $\{s_{t+k}, a_{t+k}\}_{k=1}^{\infty}$ obtained by executing π in the environment.

An important distinction from online RL is that, instead of sampling access to the environment, we assume access to a historical dataset $\mathcal{D}^{\mu}$ collected by logging experience of the policy, μ, in the form $\{o_{i,t}, a_{i,t}, r_{i,t}\}_{i=1,t=1}^{i=N,t=T}$ where, for practical purposes, the episode is truncated at T timesteps. Furthermore, we assume that the agent can only be trained on a limited collection of POMDPs $\mathcal{M}_{\text{train}} = \{M_i\}_{i=1}^{m}$, and its performance is evaluated on the set of test POMDPs $\mathcal{M}_{\text{test}}$. We assume that both $\mathcal{M}_{\text{train}}$ and $\mathcal{M}_{\text{test}}$ were sampled from a common task distribution and that every POMDP $M_i \in \mathcal{M} = \mathcal{M}_{\text{train}} \cup \mathcal{M}_{\text{test}}$ shares the same transition dynamics and reward function with $\mathcal{M}$ but has a different observation function $p_{i,\mathcal{O}}$. Importantly, since we perform control from pixels, we are in the POMDP setting [see Yarats et al., 2019] and therefore emphasize the difference between observations o_t and corresponding states s_t throughout the paper.

3.2 Representation learning

Previous works in the RL literature have studied the use of auxiliary signals to improve generalization performance. Among others, Liu et al. [2020a], Agarwal et al. [2021] define the similarity of two observations to depend on the distance between action sequences rolled out from that observation under their respective optimal policies. They achieve this by finding a latent space $\mathcal{Z} \subseteq \mathcal{S}$ in which the distance $d_{\mathcal{Z}}(z, z')$ for all $z, z' \in \mathcal{Z}$ is equivalent to distance between true latent states $d_{\mathcal{S}}(s, s')$ for all $s, s' \in \mathcal{S}$; the aforementionned works learn $\mathcal{Z}$ by optimizing action-based similarities between observations.

In practice, latent space z is decoded from observation o using a latent state decoder $f : \mathcal{O} \to \mathcal{Z}$ from observation o_t. Through the paper, we assume that all value functions have a linear form in the latent decoded state, i.e. $Q_\theta(o, a) = \theta_a^\top f_\psi(o) = \theta_a^\top z_\psi$, which agrees with our practical implementation of all algorithms. Within this model family, the ability of an RL agent to correctly decode latent states from unseen observations directly affects its policy, and therefore, its generalization capabilities. In the next section, we discuss why representation learning is important for offline RL, and how existing action-based similarity metrics fail to recover the latent states for important sets of POMDPs.

4 Motivating example

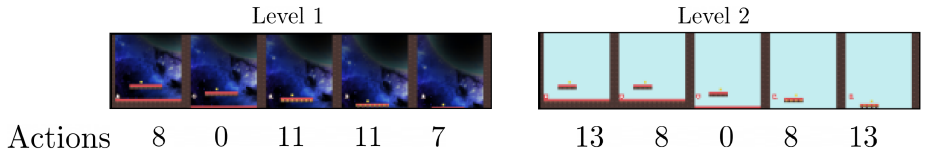

Figure 1: Two levels of the Climber game from the Procgen benchmark [Cobbe et al., 2020] with *near-identical* latent states and value functions but *drastically different* action sequences.

Multiple recently proposed self-supervised objectives [Liu et al., 2020a, Agarwal et al., 2021] conjecture that observations $o_1 \in M_1, o_2 \in M_2$ that emit similar future action sequences under optimal policies π_1^*, π_2^* should be decoded into nearby latent states z_1, z_2. While this heuristic can correctly group observations with respect to their true latent state in simple action spaces, it fails to identify similar pairs of trajectories in POMDPs with multiple optimal policies. For instance, two trajectories might visit an identical set of latent states, but have drastically different actions.

Fig. 1 shows one such example: two levels of the Climber game have a near-identical true latent state (see Appendix) and value function (average normalized mean-squared error of 0.0398 across episode), while having very different action sequences from a same PPO policy (average total variation distance of 0.4423 across episode). The problem is especially acute in Procgen, since the PPO

policy is high-entropy for some environments (see Fig. 4), i.e. various levels can have multiple drastically different near-optimal policies, and hence fail to properly capture observation similarities.

In this scenario, assigning observations to a similar latent state by value function similarity would yield a better state representation than reasoning about action similarities. In a POMDP with a different structure, grouping representations by action sequences can be optimal. So how do we unify these similarity metrics under a single framework?

In the next section, we use this insight to design a general way of improve representation learning through self-supervised learning of discounted future behavior.

5 Method

We propose to measure a generalized notion of future behavior similarity using generalized value functions, as defined by the corresponding cumulant function. The choice of cumulant determines which components of the future trajectory are most relevant for generalization.

5.1 Quantifying future behavior with GVFs

An RL agent's future discounted behavior can be quantified not only by its the value function, but other auxiliary signals, for example, by its observation occupancy measure, known as successor features [Dayan, 1993, Barreto et al., 2016]. The choice of the examined signal quantifies the properties the agent will exhibit in the future, such as accumulated returns, or observation visitation density. See Thm. 2 for the connection between successor features and interpolation error in our method.

Following the work of Sutton et al. [2011], we can broaden the class of value functions to any kind of cumulative discounted signal, as defined by a bounded cumulant function $c : \mathcal{O} \times \mathcal{A} \to \mathbb{R}^d$, s.t. $|c(o,a)| \leq c_{\max}$ for $c_{\max} = \sup_{o,a \in \mathcal{O}\times\mathcal{A}} c(o,a)$. While typically cumulants are scalar-valued functions (e.g. reward), we also make use of the vector-valued case for learning the successor features [Barreto et al., 2016], in which case the norm of $c(o,a)$ is bounded.

Definition 1 (Generalized value function) *Let c be any bounded function over $\mathbb{R}^d$, let $\gamma \in [0,1]$ and μ any policy. The generalized value function is defined as*

$$G^{\mu}(o_t) = \mathbb{E}_{\mathbb{P}_t^{\mu}}[\sum_{k=1}^{\infty} \gamma^k c(o_{t+k}, a_{t+k}) | o_t] \tag{2}$$

for any timestep $t \geq 1$ and $o_t \in \mathcal{O}$.

Since, in our case, we can learn G^{μ} for each distinct POMDP M_i for the dataset $\mathcal{D}^{\mu}$, we index the GVF using the POMDP index, i.e. $G_i^{\mu} = \texttt{LearnGVF}(c, \mathcal{D}^{\mu}, i)$[3].

Algorithm 1: $\texttt{LearnGVF}(c, \mathcal{D}^{\mu}, i, \theta^{(0)}, J, \alpha, \gamma)$: Offline estimation of GVF $\hat{G}_i^{\mu}$

Input : Cumulant function c, dataset $\mathcal{D}^{\mu}$, POMDP label i, initial parameters $\theta^{(0)}$, target parameters $\tilde{\theta}$, latent state decoder f, iterations J, learning rate α, discount γ

for $j = 1, .., J$ **do**
- $o, a, o' \sim \mathcal{D}[i]$; `// Sample transition from POMDP` i
- $c \leftarrow c(o,a)$;
- $o \leftarrow$ random crop(o);
- $z, z' \leftarrow f(o), f(o')$;
- $\theta^{(j)} \leftarrow \theta^{(j-1)} - \alpha \nabla_{\theta^{(j-1)}} (G_{\theta^{(j-1)}}(z) - c - \gamma G_{\tilde{\theta}^{(j-1)}}(z'))^2$;
- Update target parameters $\tilde{\theta}$ with β of online parameters θ;

5.2 Measuring distances between GVFs of different POMDPs

Examining the difference between future behaviors of two observations quantifies the exact amount of expected behavior change between these two observations. Using the GVF framework, we could

[3] In practice, the learning is parallelized

compute the distance between $o_1 \in M_1$ and $o_2 \in M_2$ by first estimating the latent state with $z = f(o)$ using a (learned) latent state decoder f, and then evaluating the distance

$$d_\mu({o_1}^i, {o_2}^j) = |G_i^\mu(f(o_1)) - G_j^\mu(f(o_2))| \;\; i, j = 1, 2, .., \tag{3}$$

a measure of dissimilarity that can then be used in a contrastive loss.

However, the distance between GVFs from two different POMDPs can have drastically different scales: $|G_1^\mu(o_1) - G_2^\mu(o_2)| \leq \frac{c_{1,\max}^\mu + c_{2,\max}^\mu}{1-\gamma}$, making point-wise comparison meaningless. The issue is less acute for cumulants which induce a unnormalized density estimate (e.g. indicator functions for successor representation), and more problematic when the cumulant incorporates the extrinsic reward function. To avoid this problem, we suggest performing a comparison based on order statistics.

A robust distance estimate between GVF signals across POMDPs can be obtained by looking at the cumulative distribution function of G_i denoted $F_i(g) = \mathbb{P}[G_i(o_t) \leq g]$ for all $o_t \in \mathcal{O}$. G_i is a deterministic GVF with the set of discontinuity points of measure 0, and as such F_i can be understood through the induced state distribution $\mathbb{P}_t^\mu$ (using continuous mapping theorem from Mann and Wald [1943]). It can be estimated from n independent, identically distributed samples of $\mathcal{D}^\mu$ as

$$\hat{F}_i(g) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{G_i < g}, G_i = \texttt{LearnGVF}(c, \mathcal{D}^\mu, i), \;\; g \in \left[-\frac{c_{i,\max}}{1-\gamma}, \frac{c_{i,\max}}{1-\gamma} \right] \tag{4}$$

and its inverse, the empirical quantile function [van der Vaart, 1998]

$$\hat{F}_i^{-1}(p) = \inf\{g \in \left[-\frac{c_{i,\max}^\mu}{1-\gamma}, \frac{c_{i,\max}^\mu}{1-\gamma} \right] : p \leq F_i(g)\}, \; p \in [0, 1] \tag{5}$$

We use the empirical quantile function to partition the range of all GVFs into K quantile bins, i.e. disjoint sets with identical size where the set corresponding to quantile k is defined as $I_i(k) = \{o \in M_i : F_i^{-1}(\frac{k}{K}) \leq G_i^\mu(o) \leq F_i^{-1}(\frac{k+1}{K})\}$ and it's aggregated version as $I(k) = \cup_{i=1}^m I_i(k)$.

Importantly, we augment the dataset $\mathcal{D}^\mu$ with observation-specific labels, which correspond to the index of the quantile bin into which the GVF G of an observation $o \in M_i$ falls into:

$$l_i(o) = \max_k \mathbb{1}_{o \in I_i(k)} \tag{6}$$

These self-supervised labels are then used in a multiclass InfoNCE loss [Oord et al., 2018], which is a variation of metric learning with respect to the quantile distance defined above [Khosla et al., 2020, Song and Ermon, 2020].

5.3 Self-supervised learning of GSFs

After augmenting the offline dataset with observation labels, we use a simple self-supervised learning procedure to minimize distance in the latent representation space between observations with identical labels.

First, the observation o is encoded using a non-linear encoder $f_\psi : \mathcal{O} \to \mathcal{Z}$ with parameters ψ into a latent state representation $z = f_\psi(o)$[4]. The representation z is then passed into two separate trunks: 1) a linear matrix θ_a which recovers the state-action value function $Q_\theta(o, a) = \theta_a^\top z$, and 2) a non-linear projection network $h_\theta : \mathcal{Z} \to \mathcal{Z}$ with parameters θ_h to obtain a new embedding, used for contrastive learning.

The projection $h(z)$ is then used used in a multiclass InfoNCE loss [Oord et al., 2018, Song and Ermon, 2020] where a linear classifier $\mathbf{W} \in \mathbb{R}^{|\mathcal{Z}| \times K}$ aims to correctly predict the observation labels (i.e. quantile bins $k = 1, 2, .., K$) from $h(z)$:

$$\ell_{\text{NCE}}(\theta_h, \psi, \mathbf{W}) = -\mathbb{E}_{o \sim \mathcal{D}^\mu}\left[\sum_{k=1}^{K} \mathbb{1}_{l(o)=k} \text{LogSoftmax}[\mathbf{W}^\top h(f_\psi(o))/\tau]_k \right], \tag{7}$$

[4]This encoder is different from the one used to evaluate the GVFs.

where $\tau > 0$ is a temperature parameter.

Our empirical findings suggest that this version of the loss is more stable than other multi-class contrastive losses (see Appendix 7.3).

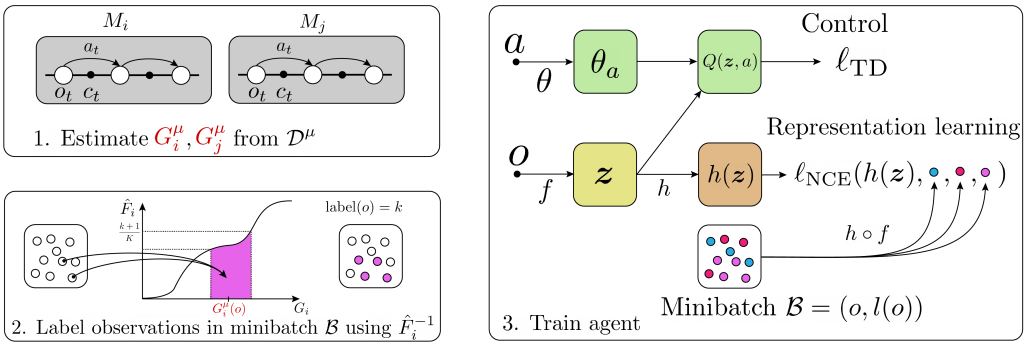

Figure 2: Schematic view of GSF : the offline dataset $\mathcal{D}^\mu$ is used to estimate POMDP-specific GVFs wrt some cumulant function c, whose quantiles are then used to label each observation in the dataset. These labels are then used in a multi-class contrastive learning procedure along with offline RL learning.

5.4 Algorithm

Our method relies on the approximation oracle `LearnGVF`, to produce GVF estimates, later used in the contrastive learning phase.

Since we are concerned with the offline RL setting, we add our auxiliary loss on top of Conservative Q-learning [CQL, Kumar et al., 2020], a strong baseline. CQL is trained using a linear combination of Q-learning [Watkins and Dayan, 1992, Ernst et al., 2005] and behavior-regularization:

$$\ell_{\text{CQL}}(\theta) = \mathbb{E}_{o,a,r,o' \sim \mathcal{D}^\mu}[(r + \gamma \max_{a' \in \mathcal{A}} Q_{\tilde{\theta}}(o', a') - Q_\theta(o,a))^2] + \lambda \mathbb{E}_{s \sim \mathcal{D}^\mu}[LSE(Q_\theta(o,a)) - \mathbb{E}_{a \sim \mu}[Q_\theta(o,a)]], \tag{8}$$

for $\lambda \geq 0$, $\tilde{\theta}$ target network parameters[5] and LSE being the log-sum-exp operator [6].

Algorithm 2: GSF : Offline RL with future behavior observation matching

Input : Dataset $\mathcal{D} \sim \mu$, initialized Q-function Q_θ with encoder f_{θ_f} and action weights θ_a, per-POMDP set of GVFs $\mathcal{G} = \{G_i^\mu\}_{i \in}$, state projection network h_ψ, epoch number J, number of POMDPs m, number of quantiles K, temperature parameter τ, exponential moving average parameter β

for *epoch* $j = 1, 2, .., J$ **do**
  for *minibatch* $\mathcal{B} \sim \mathcal{D}$ **do**
    `/* Data augmentation on observation */`
    $o \leftarrow$ random crop(o) for all $o \in \mathcal{B}$;
    $z \leftarrow f_\psi(o)$ for all $o \in \mathcal{B}$;
    `/* Update CQL agent */`
    Update θ_a, ψ using $\nabla_{\theta_a, \psi} \ell_{\text{CQL}}(\theta)$;
    `/* Compute` $\mathcal{G}$ `quantiles */`
    for *POMDP* $M_i = 1, 2, .., m$ **do**
      Estimate $\hat{F}_i^{-1}$ of G_i^μ from $\mathcal{B}$;
      for *observation* $o \in \mathcal{B} \cap M_i$ **do**
        $l(o) \leftarrow k$ if $\hat{F}_i^{-1}(\frac{k}{K}) \leq G_i^\mu(o) \leq \hat{F}_i^{-1}(\frac{k+1}{K})$;
    `/* Update encoder and projection network */`
    Update $\theta_h, \psi, \mathbf{W}$ using $\nabla_{\theta_h, \psi, \mathbf{W}} \ell_{\text{NCE}}(\theta_h, \psi, \mathbf{W})$ computed with $z, l(o)$ and τ ;
    Update CQL agent's target network with β of online parameters ψ, θ;

[5] A copy of θ updated solely using an exponential moving average (see Appendix).
[6] https://en.wikipedia.org/wiki/LogSumExp

Alg. 2 summarizes the learning procedure for GSF as implemented on top of a CQL agent for a discrete action space. In our experiments, all baselines use random crops as data augmentation.

Connection to existing methods Our framework is able to recover objectives similar to those of prior works by carefully designing the cumulant function.

- **Cross-State Self-Constraint [CSSC, Liu et al., 2020a]**: In CSSC, observations o_1, o_2 are considered similar if they have identical future action sequences of length K under some fixed policy; a total of $|\mathcal{A}|^K$ distinct classes are possible. This approach can be approximated in our framework by picking $c(o_t, a_t) = \mathbb{1}_{a_t}(a),\ \forall a \in \mathcal{A}$. The problem reduces to a $|\mathcal{A}|^{T-t}$-way classification problem for observations of timestep t, which GSF approximates using K quantiles.
- **Policy similarity embedding [PSE, Agarwal et al., 2021]**: PSEs balance the distance between local optimal behaviors and long-term dependencies in the transitions, notably using d_{TV}. If we consider the space of Boltzmann policies $\pi_{\text{Boltzmann}}$ with respect to an POMDP-specific value function Q, then choosing $c(o_t, a_t) = r(s_t, a_t)$ in GSF will effectively compute the distance between unnormalized policies.

5.5 Choice of number of quantiles K

How should the number of quantiles K be set, and what is the effect of smaller/ larger values of K on the observation distance? Thm. 1 highlights a trade-off when choosing the number of quantiles bins empirically.

Theorem 1 *Let G_1, G_2 be generalized value functions with cumulants c_1, c_2 from respective POMDPs M_1, M_2, K be the number of quantile bins, n_1, n_2 the number of sample transitions from each POMDP. Suppose that $\mathbb{P}[\sup_{t=1,2,..} |c_1(o_{1,t}, \mu(o_{1,t})) - c_2(o_{2,t}, \mu(o_{2,t}))| > {}^{(1-\gamma)\varepsilon}/_{\gamma}] \leq \delta$. Then, for any $k = 1, 2, .., K$ and $\varepsilon > 0$ the following holds without loss of generality:*

$$\mathbb{P}\Big[\sup_{o_1, o_2 \in I(k)} |G_1(o_1) - G_2(o_2)| > 3\varepsilon\Big] \leq 2e^{-2n_1\varepsilon^2/4} + p(n_1, K, \varepsilon) + \delta \tag{9}$$

where

$$p(n, K, \varepsilon) = \mathbb{P}\Big[\sup_{k=1,2,..,K} \left|\hat{F}_1^{-1}\left({}^{k+1}/_{K}\right) - \hat{F}_1^{-1}\left({}^{k}/_{K}\right)\right| > \varepsilon\Big] \tag{10}$$

.

The proof can be found in the Appendix Sec. 7.2. For POMDP M_1, the error decreases monotonically with increasing bin number K (second term) but the variance of bin labels depends on the number of sample transitions n_1 (first term). The inter-POMDP error (third term) does not affect the bin assignment. Hence, choosing a large K will amount to pairing states by rankings, but results in high variance, as orderings are estimated from data and each bin will have $n = 1$. Setting K too small will group together unrelated observations, inducing high bias.

6 Experiments

Unlike for single task offline RL [Fu et al., 2020], most works on zero-shot generalization from offline data either come up with an *ad hoc* solution suiting their needs, e.g. [Ball et al., 2021], or assess performance on benchmarks that do not evalute generalization across observation functions [e.g., Yu et al., 2020]. To accelerate progress in this field, we devised the offline Procgen benchmark, an offline RL dataset to directly test for generalization of offline RL agents across observation functions[7].

Offline Procgen benchmark We evaluate the proposed approach on an offline version of the Procgen benchmark [Cobbe et al., 2020], which is widely used to evaluate zero-shot generalization across complex visual perturbations. Given a random seed, Procgen allows to sample procedurally generated level configurations for 16 games under various complexity modes: "easy", "hard" and "exploration". The dataset is obtained as follows: we first pre-train a PPO [Schulman et al., 2017] agent for 25M timesteps on 200 levels of "easy" distribution for each environment[8] ("easy" mode is

[7] The benchmark will be open-sourced.

[8] We use the TFAgents' implementation [Guadarrama et al., 2018]

widely used to test generalization capabilities [Cobbe et al., 2020, Raileanu and Fergus, 2021, Mazoure et al., 2021a]). All agents use the IMPALA encoder architecture [Espeholt et al., 2018], which has enough parameters to allow better generalization performance, compared to other models [e.g., Mnih et al., 2015].

Results We compare the zero-shot performance on the entire distribution of "easy" POMDPs for GSF against that of strong RL and representation learning baselines: behavioral cloning (BC) - to assess the quality of the PPO policy, CQL [Kumar et al., 2020] - the current state-of-the-art on multiple offline benchmarks which balances RL and BC objectives, CURL [Srinivas et al., 2020], CTRL [Mazoure et al., 2021a], DeepMDP [Gelada et al., 2019] - which learns a metric closely related to bisimulation across the MDP, Value Prediction Network [VPN, Oh et al., 2017] - which combines model-free and model-based learning of values, observations, next observations, rewards and discounts, Cross-State Self-Constraint [CSSC, Liu et al., 2020a] - which boosts similarity of observations with identical action sequences, as well as Policy Similarity Embeddings [Agarwal et al., 2020], which groups observation representations based on distance in optimal policy space.

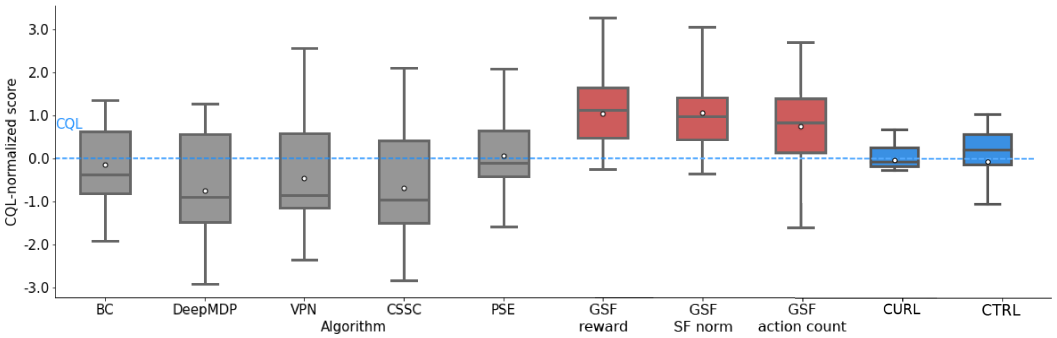

Figure 3: Returns on the offline Procgen benchmark [Cobbe et al., 2020] after 1M training steps. Boxplots are constructed over 5 random seeds and all 16 games; each method is normalized by the per-game median CQL performance. White dots represent average of distribution.

Fig. 3 shows the performance of all methods over 5 random seeds and all 16 games on the offline Procgen benchmark after 1 million training steps. Per-game average scores for all methods can be found in Tab. 2 (Appendix). The scores are standardized per-game using the downstream task's (offline RL) performance, in this case implemented by CQL. It can be seen that GSF performs better than other offline RL and representation learning baselines.

Using different cumulants functions can lead to different label assignments and hence different similarity groups. Fig. 3 examines the performance of GSF with respect to 3 cumulants: 1) $r(s_t, a_t)$, rewards s.t. GSF learns the policy's Q^μ-value, 2) $\mathbb{1}_{o_t}(o)$, the successor representation[9] [Dayan, 1993, Barreto et al., 2016] s.t. GSF learns induced distribution over $\mathcal{D}^\mu$ [Machado et al., 2020] and 3) $\mathbb{1}_{a_t}(a)$, action counts, s.t. GSF learns discounted policy. While rewards and successor feature cumulant choices leads to similar performance, using action-based distance leads to larger variance.

7 Discussion

In this work we proposed GSF, a novel algorithm which combines reinforcement learning with representation learning to improve zero-shot generalization performance on challenging, pixel-based control tasks. GSF relies on computing the similarity between observation pairs with respect to any instantaneous accumulated signal, which leads to improved empirical performance on the newly introduced offline Procgen benchmark. Theoretical results suggest that GSF 's hyperparameter choice depends on a trade-off between finite sample approximation and extrapolation error.

While our work answered some questions regarding zero-shot generalization in offline RL, some questions persist: can GVF-based distances be included in a contrastive objective without the need

[9] In the continuous observation space, we learn a d-dimensional successor feature vector z_ψ via TD and computing the quantiles over $||z_\psi||_1$.

for quantile discretization (perhaps through re-scaling or order statistics)? Can the cumulant function be chosen *a priori* for a specific task structure other than Procgen, and shown to lead to optimal representations?

References

A. Agarwal, M. Henaff, S. Kakade, and W. Sun. Pc-pg: Policy cover directed exploration for provable policy gradient learning. *Neural Information Processing Systems*, 2020.

R. Agarwal, M. C. Machado, P. S. Castro, and M. G. Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. *arXiv preprint arXiv:2101.05265*, 2021.

P. J. Ball, C. Lu, J. Parker-Holder, and S. Roberts. Augmented world models facilitate zero-shot dynamics generalization from a single offline environment. *International Conference on Machine Learning*, 2021.

A. Barreto, W. Dabney, R. Munos, J. J. Hunt, T. Schaul, H. Van Hasselt, and D. Silver. Successor features for transfer in reinforcement learning. *arXiv preprint arXiv:1606.05312*, 2016.

P. L. Bartlett. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE transactions on Information Theory*, 44(2):525–536, 1998.

M. G. Bellemare, S. Candido, P. S. Castro, J. Gong, M. C. Machado, S. Moitra, S. S. Ponda, and Z. Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, 2020.

J. Boyan and A. W. Moore. Generalization in reinforcement learning: Safely approximating the value function. *Advances in neural information processing systems*, pages 369–376, 1995.

M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

P. S. Castro. Scalable methods for computing state similarity in deterministic markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10069–10076, 2020.

P. S. Castro and D. Precup. Using bisimulation for policy transfer in mdps. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

A. S. Chen, S. Nair, and C. Finn. Learning generalizable robotic reward functions from" in-the-wild" human videos. *arXiv preprint arXiv:2103.16817*, 2021.

C.-A. Cheng, A. Kolobov, and A. Swaminathan. Heuristic-guided reinforcement learning. *arXiv preprint arXiv:2106.02757*, 2021.

K. Cobbe, O. Klimov, C. Hesse, T. Kim, and J. Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pages 1282–1289. PMLR, 2019.

K. Cobbe, C. Hesse, J. Hilton, and J. Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pages 2048–2056. PMLR, 2020.

P. Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

A. Dvoretzky, J. Kiefer, and J. Wolfowitz. Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *The Annals of Mathematical Statistics*, pages 642–669, 1956.

D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416. PMLR, 2018.

N. Ferns, P. Panangaden, and D. Precup. Metrics for finite markov decision processes. In *UAI*, volume 4, pages 162–169, 2004.

J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

S. Fujimoto, D. Meger, and D. Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062. PMLR, 2019.

C. Gelada, S. Kumar, J. Buckman, O. Nachum, and M. G. Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning*, pages 2170–2179. PMLR, 2019.

J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

S. Guadarrama, A. Korattikara, O. Ramirez, P. Castro, E. Holly, S. Fishman, K. Wang, E. Gonina, N. Wu, E. Kokiopoulou, L. Sbaiz, J. Smith, G. Bartók, J. Berent, C. Harris, V. Vanhoucke, and E. Brevdo. TF-Agents: A library for reinforcement learning in tensorflow. https://github.com/tensorflow/agents, 2018. URL https://github.com/tensorflow/agents. [Online; accessed 4-October-2021].

K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

M. Hessel, H. Soyer, L. Espeholt, W. Czarnecki, S. Schmitt, and H. van Hasselt. Multi-task deep reinforcement learning with popart. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3796–3803, 2019.

I. Higgins, A. Pal, A. Rusu, L. Matthey, C. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner. Darla: Improving zero-shot transfer in reinforcement learning. In *International Conference on Machine Learning*, pages 1480–1490. PMLR, 2017.

R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

N. K. Jong and P. Stone. State abstraction discovery from irrelevant state variables. In *IJCAI*, volume 8, pages 752–757. Citeseer, 2005.

P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan. Supervised contrastive learning. *Neural Information Processing Systems*, 2020.

R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951*, 2020.

I. Kostrikov, D. Yarats, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

I. Kostrikov, J. Tompson, R. Fergus, and O. Nachum. Offline reinforcement learning with fisher divergence critic regularization. *arXiv preprint arXiv:2103.08050*, 2021.

A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

S. Lange, T. Gabel, and M. Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer, 2012.

L. Li, T. J. Walsh, and M. L. Littman. Towards a unified theory of state abstraction for mdps. *ISAIM*, 4:5, 2006.

Z. Lin, D. Yang, L. Zhao, T. Qin, G. Yang, and T.-Y. Liu. Reward decomposition with representation decomposition. *Advances in Neural Information Processing Systems*, 33, 2020.

G. T. Liu, P.-J. Cheng, and G. Lin. Cross-state self-constraint for feature generalization in deep reinforcement learning. 2020a.

L. Liu, W. Hamilton, G. Long, J. Jiang, and H. Larochelle. A universal representation transformer layer for few-shot image classification. *arXiv preprint arXiv:2006.11702*, 2020b.

Y. Liu, A. Swaminathan, A. Agarwal, and E. Brunskill. Provably good batch reinforcement learning without great exploration. *NeurIPS*, 2020c.

M. C. Machado, M. G. Bellemare, and M. Bowling. Count-based exploration with the successor representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5125–5133, 2020.

H. B. Mann and A. Wald. On stochastic limit and order relationships. *The Annals of Mathematical Statistics*, 14(3):217–226, 1943.

B. Mazoure, R. T. d. Combes, T. Doan, P. Bachman, and R. D. Hjelm. Deep reinforcement and infomax learning. *Neural Information Processing Systems*, 2020.

B. Mazoure, A. M. Ahmed, P. MacAlpine, R. D. Hjelm, and A. Kolobov. Cross-trajectory representation learning for zero-shot generalization in rl. *arXiv preprint arXiv:2106.02193*, 2021a.

B. Mazoure, P. Mineiro, P. Srinath, R. S. Sedeh, D. Precup, and A. Swaminathan. Improving long-term metrics in recommendation systems using short-horizon offline rl. *arXiv preprint arXiv:2106.00589*, 2021b.

D. Misra, M. Henaff, A. Krishnamurthy, and J. Langford. Kinematic state abstraction and provably efficient rich-observation reinforcement learning. In *International conference on machine learning*, pages 6961–6971. PMLR, 2020.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

K. P. Murphy. A survey of pomdp solution techniques. *environment*, 2:X3, 2000.

O. Nachum and M. Yang. Provable representation learning for imitation with contrastive fourier features. *Neural Information Processing Systems*, 2021.

J. Oh, S. Singh, and H. Lee. Value prediction network. *arXiv preprint arXiv:1707.03497*, 2017.

A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

R. Raileanu and R. Fergus. Decoupling value and policy for generalization in reinforcement learning. *International Conference on Machine Learning*, 2021.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

M. Schwarzer, A. Anand, R. Goel, R. D. Hjelm, A. Courville, and P. Bachman. Data-efficient reinforcement learning with self-predictive representations. *International Conference on Learning Representations*, 2020.

M. Schwarzer, N. Rajkumar, M. Noukhovitch, A. Anand, L. Charlin, D. Hjelm, P. Bachman, and A. Courville. Pretraining representations for data-efficient reinforcement learning. *arXiv preprint arXiv:2106.04799*, 2021.

S. Sinha and A. Garg. S4rl: Surprisingly simple self-supervision for offline reinforcement learning. *arXiv preprint arXiv:2103.06326*, 2021.

J. Song and S. Ermon. Multi-label contrastive predictive coding. *Neural Information Processing Systems*, 2020.

X. Song, Y. Jiang, S. Tu, Y. Du, and B. Neyshabur. Observational overfitting in reinforcement learning. *International Conference on Learning Representations*, 2019.

A. Srinivas, M. Laskin, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *International Conference on Machine Learning*, 2020.

A. Stone, O. Ramirez, K. Konolige, and R. Jonschkowski. The distracting control suite–a challenging benchmark for reinforcement learning from pixels. *arXiv preprint arXiv:2101.02722*, 2021.

A. Stooke, K. Lee, P. Abbeel, and M. Laskin. Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning*, pages 9870–9879. PMLR, 2021.

W. Sun, J. A. Bagnell, and B. Boots. Truncated horizon policy search: Combining reinforcement learning & imitation learning. *arXiv preprint arXiv:1805.11240*, 2018.

R. S. Sutton, J. Modayil, M. Delp, T. Degris, P. M. Pilarski, A. White, and D. Precup. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 761–768, 2011.

A. Swaminathan and T. Joachims. Batch learning from logged bandit feedback through counterfactual risk minimization. *The Journal of Machine Learning Research*, 16(1):1731–1755, 2015.

A. Touati and Y. Ollivier. Learning one representation to optimize all rewards. *arXiv preprint arXiv:2103.07945*, 2021.

E. Triantafillou, T. Zhu, V. Dumoulin, P. Lamblin, U. Evci, K. Xu, R. Goroshin, C. Gelada, K. Swersky, P.-A. Manzagol, et al. Meta-dataset: A dataset of datasets for learning to learn from few examples. *International Conference on Learning Representations*, 2019.

G. Valle-Pérez and A. A. Louis. Generalization bounds for deep learning. *arXiv preprint arXiv:2012.04115*, 2020.

A. W. van der Vaart. Asymptotic statistics. cambridge series in statistical and probabilistic mathematics, 1998.

R. Wang, Y. Wu, R. Salakhutdinov, and S. M. Kakade. Instabilities of offline rl with pre-trained neural representation. *arXiv preprint arXiv:2103.04947*, 2021a.

Y. Wang, R. Wang, and S. M. Kakade. An exponential lower bound for linearly-realizable mdps with constant suboptimality gap. *arXiv preprint arXiv:2103.12690*, 2021b.

C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Y. Wu, G. Tucker, and O. Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus. Improving sample efficiency in model-free reinforcement learning from images. *arXiv preprint arXiv:1910.01741*, 2019.

T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pages 1094–1100. PMLR, 2020.

A. Zhang, R. McAllister, R. Calandra, Y. Gal, and S. Levine. Learning invariant representations for reinforcement learning without reconstruction. *International Conference on Learning Representations*, 2020.

Appendix

7.1 Experimental details

Name	Description	Value
n_{MDPs}	Number of training POMDPs in the dataset	200
γ	Discount factor	0.99
Batch size	Batch size	1024
Hidden layers	Size of Q critic post-encoder layers	$256 \times 256 \times \lvert\mathcal{A}\rvert$
λ	CQL regularization coefficient	1
Data augmentation	Type of data augmentation use in all methods	Random crop
α	Target soft update rate	0.005
Hidden layers	Size of G network post-encoder layers	$256 \times 256 \times \lvert\mathcal{A}\rvert * n_{\text{MDPs}}$
G pretraining steps	Number of iterations to pretrain GVF estimator for	100,000
K	Number of quantiles	7
τ	Softmax temperature	0.5

Table 1: Experiments' parameters

For improved time efficiency, we jointly train the estimators $G_1, G_2, .., G_m$ by first decoding each observation $o \in M_i$ into a latent representation $z = f_{\psi'}(o)$ and then passing the representation through a non-linear network $h' : \mathcal{Z} \to \mathbb{R}^{d_c \times m}$ where d_c is the dimensionality of the cumulant function's output. The output of h' is then split into m disjoint chunks, with chunk i used in the temporal difference loss ℓ_{TD} with the reward replaced by the cumulant function. However, due to different scales of cumulants across POMDPs, we use the PopArt re-normalization scheme [Hessel et al., 2019] to prevent gradient instability. All GVFs are trained for $50k$ iterations using ℓ_{TD}.

Data augmentation was taken to be solely random crops, since this type of data augmentation was shown to be sufficient to improve performance of deep RL agents in pixel-based control tasks [Kostrikov et al., 2020]. To do so, we first symmetrically padded the $64 \times 64 \times 3$ observation up to $68 \times 68 \times 3$, and applied a 64×64 random crop on the resulting tensor.

All baselines' code was taken from their respective repositories and adapted into our codebase (except DeepMDP which had to be implemented from third-party sources).

Hyperparameter search: We allowed each method to tune one hyperparameter due to computational budget constraints. For CQL, λ was tuned and found to be 1. For GSFs, we tuned K, the number of quantiles. For all other methods, we tuned the auxiliary loss coefficient(s). Performance of GSFs vs some hyperparameter choices can be seen in Fig. 6.

Dataset composition We first pre-trained PPO [Guadarrama et al., 2018] for 25M frames for all 16 games. Then, we conducted rollouts with the resulting policy under an ε_t-greedy strategy; ε_t was allowed to decay according to the rule

$$\varepsilon_t = 0.1 - 3.96 \times 10^{-9} t,$$

which has two endpoints: $\varepsilon_0 = 0.1$ and $\varepsilon_{25M} = 0$. This was done in order to prevent collapse of the Q-values due to the log-sum-exp term in CQL to very low negative values (if action coverage is not sufficient in the dataset).

Target updates: As is common in multiple deep RL algorithms [Kostrikov et al., 2020, 2021], the bootstrap target estimates are typically computed using a value network with parameters $\tilde{\theta}$, which are in turn periodically updated using an exponential moving average of historical values with the current online network parameters θ with update rate α:

$$\tilde{\theta} = \alpha\theta + (1 - \alpha)\tilde{\theta} \tag{11}$$

7.2 Proofs

Proof 1 (Thm. 1) *First, consider some arbitrary quantile bin $k = 1, 2, .., K$.*

$$\begin{aligned}\sup_{o_1,o_2\in I(k)} |G_1^\mu(o_1) - G_2^\mu(o_2)| &= \sup_{o_1,o_2\in I(k)} |G_1^\mu(o_1) - G_1^\mu(o_2) + G_1^\mu(o_2) - G_2^\mu(o_2)| \\ &\leq \sup_{o_1,o_2\in I(k)} |G_1^\mu(o_1 - o_2)| + |G_1^\mu(o_2) - G_2^\mu(o_2)|\end{aligned} \tag{12}$$

Since the cumulant is, in practice, estimated from $\mathcal{D}^\mu$, it follows that $c(s,a) \in [-c^\mu_{i,max}, c^\mu_{i,max}] \subseteq [-c_{max}, c_{max}]$ for all $s, a \in \mathcal{S}, \mathcal{A}$. Since the disparity between cumulants for POMDP M_1, M_2 comes from the uneven coverage by μ, we can denote this as $\delta_{1,2}(t) = |c(o_{1,t}, \mu(o_{1,t})) - c(o_{2,t}, \mu(o_{2,t}))|$.

Suppose that $\mathbb{P}[\sup_{k=1,2,..} \delta_{1,2}(t+k) > \varepsilon] \leq p^\mu(c_1, c_2, \varepsilon)$. Then,

$$\begin{aligned}\mathbb{P}[|G_1^\mu(o_t) - G_2^\mu(o_t)| > \varepsilon] &= \mathbb{P}[\mathbb{E}_{\mathbb{P}_t^\mu}[\sum_{k=1}^{\infty} \gamma^k |c_i^\mu(o_{t+k}, a_{t+k}) - c_j^\mu(o_{t+k}, a_{t+k})||o_t] > \varepsilon] \\ &= \mathbb{P}[\mathbb{E}_{\mathbb{P}_t^\mu}[\sum_{k=1}^{\infty} \gamma^k \delta_{1,2}(t+k)|o_t] > \varepsilon] \\ &\leq \mathbb{P}[\sup_{k=1,2,..,K} \delta_{1,2}(t+k) \mathbb{E}_{\mathbb{P}_t^\mu}[\sum_{k=1}^{\infty} \gamma^k |o_t] > \varepsilon] \\ &\leq \mathbb{P}[\sup_{k=1,2,..,K} \delta_{1,2}(t+k) > \frac{(1-\gamma)\varepsilon}{\gamma}] \\ &\leq p^\mu(c_1, c_2, {}^{(1-\gamma)\varepsilon}/_{\gamma})\end{aligned} \tag{13}$$

since M_1, M_2 share the same induced distribution $\mathbb{P}_t^\mu$. Due to stationarity, we can drop the time index.

Now, the first term can be decomposed with the empirical distribution function $\hat{F}_i^{-1}$ estimated from n_i samples of POMDP M_i:

$$\begin{aligned}\left|F_1^{-1}\left(\frac{k+1}{K}\right) - F_1^{-1}\left(\frac{k}{K}\right)\right| &\leq \sup_{k'=1,2,..,K} \left|F_1^{-1}\left(\frac{k'+1}{K}\right) - F_1^{-1}\left(\frac{k'}{K}\right)\right| \\ &\leq \sup_{k'=1,2,..,K} \Delta(F_1, n_1, \frac{k'+1}{K}) + \Delta(F_1, n_1, \frac{k'}{K}) + \Delta(F_1, n_1, \frac{k'+1}{K}, \frac{k'}{K}) \\ &\leq 2 \sup_{k'=1,2,..,K} \Delta(F_1, n_1, \frac{k'}{K}) + \sup_{k'=1,2,..,K} \Delta(F_1, n_1, \frac{k'+1}{K}, \frac{k'}{K})\end{aligned} \tag{14}$$

where

$$\Delta(F, n, p) = \left|F^{-1}(p) - \hat{F}^{-1}(p)\right|$$

and

$$\Delta(F, n, p_1, p_2) = \left|\hat{F}^{-1}(p_1) - \hat{F}^{-1}(p_2)\right|,$$

and dependence of F on n is implicit.

We now use the union bound to observe the fact that $\mathbb{P}[\sum_{i=1}^n |X_i| > n\varepsilon] \leq \sum_{i=1}^n \mathbb{P}[|X_i| > \varepsilon]$ for $X_1, .., X_n$ real-valued random variables and $\varepsilon > 0$.

Using this fact, and that events listed in Eq. 14 form an increasing sequence of supersets

$$\begin{aligned}\mathbb{P}\left[\left|F_1^{-1}\left(\frac{k+1}{K}\right) - F_1^{-1}\left(\frac{k}{K}\right)\right| > 2\varepsilon\right] &\leq \mathbb{P}\left[\sup_{k'\in[0,1]} \Delta(F_1, n_1, \frac{k'}{K}) > \frac{\varepsilon}{2}\right] + \mathbb{P}\left[\sup_{k'=1,2,..,K} \Delta(F_1, n_1, \frac{k'+1}{K}, \frac{k'}{K}) > \varepsilon\right] \\ &\leq 2e^{-2n_1\varepsilon^2/4} + p(n_1, K, \varepsilon)\end{aligned} \tag{15}$$

Here, we used the known results of convergence of the empirical distribution function $\hat{F}$ to the true distribution function F as $n \to \infty$ [Dvoretzky et al., 1956]. Using the continuous mapping theorem [Mann and Wald, 1943] for a transformation with a set of discontinuities of measure 0, we transposed this result onto the empirical quantile function F^{-1}.

Since the error is monotonic in n, we symmetrize the bound by replacing n_1 by $\min(n_1, n_2)$, so that both M_1 and M_2 can be interchanged.

Theorem 2 *Let f be an encoder such that $\nabla \ell_{NCE}(\theta_h, \psi, \mathbf{W}) \approx 0$. If $c(o, a) = \mathbb{1}_{o_t}(o)$, then $I(K)$ is the set which has the smallest interpolation error over f and $I(1)$ has the largest interpolation error over f.*

Proof 2 (Thm. 2) *If $\nabla \ell_{NCE} \approx 0$, then, for all $k = 1, 2, .., K$*

$$\sup_{o_1, o_2 \in I(k)} ||G_1(o_1) - G_2(o_2)|| < \varepsilon_G \implies \sup_{o_1, o_2 \in I(k)} ||f(o_1) - f(o_2)|| < \varepsilon_f \tag{16}$$

for $\varepsilon_G, \varepsilon_f > 0$. For the specific choice of cumulant being the state indicator function, the following result is due to Machado et al. [2020]:

$$\frac{\gamma}{n_1(o) + 1} + 1 + \gamma \leq ||G(o)||_1 \leq \frac{\gamma}{n_1(o) + 1} + \frac{\gamma^2}{1 - \gamma} + 1 + \gamma, \tag{17}$$

where $n_i(o)$ is the number of times observation o was visited by policy μ in POMDP M_i (here, G is a vector-valued function). Therefore,

$$\sup_{o_1, o_2 \in I(k)} ||n_1(o_1) - n_2(o_2)|| < \varepsilon_G \implies \sup_{o_1, o_2 \in I(k)} ||f(o_1) - f(o_2)|| < \varepsilon_f, \tag{18}$$

which means that observations assigned with to similar representations have similar state visitation frequencies under μ. The set $I(K)$ has the observations with highest counts, i.e. lowest interpolation error, while $I(1)$ has the largest interpolation error.

7.3 Additional results

True latent state similarity in Procgen The true latent state in Procgen consists of a byte vector describing the entire memory state of the environment and the agent. This vector can be extracted by using the command `list(env.callmethod("get_state"))` in Python. To assess the distance between latent state vectors, we computed the negative cosine similarity between them. Fig. 1 shows two pairs of trajectories, for which the average cosine similarity between true latent states across timesteps was 0.897.

Value-function similarity in Procgen Fig. 1 shows two pairs of trajectories with drastically dissimilar (discrete) action sequences. Computing the distance between them will result in a large quantity which doesn't necessarily decrease with latent state similarity. On the other hand, the values for the first sequence of states are 8.20, 8.05, 8.74, 9.06, 9.29, and 8.35, 8.30, 8.26, 8.55, 8.40 for the second sequence of states. We can see that values are much more similar in this coupling of tasks than actions (with respect to, e.g. ℓ_1) and, if we group the states by their corresponding value magnitude (or quantiles), the encoder will learn to assign all states which look like the sequences above to a neighboring latent representation, and hence, will allow better zero-shot generalization as it will learn to ignore backgrounds, platform length and small variations in agent's position.

Entropy of PPO policies on Procgen While in some domains, policies can achieve optimality without much exploration, the Procgen benchmark requires PPO to have a non-zero entropy-boosting term (otherwise, results are suboptimal).

When the logging policies are high-entropy, many action sequences can possibly lead to high rewards. However, this does not imply that the observations in those sequences must have high similarity in latent space.

		RL+Bisim.	RL+value pred.	RL+action dist.		RL+GVF dist.
Env	BC	DeepMDP [Gelada et al., 2019]	VPN [Oh et al., 2017]	CSSC [Liu et al., 2020a]	PSE [Agarwal et al., 2021]	GSF (reward)
bigfish	-0.443686	-0.296928	0.296928	0.116041	0.189761	0.153584
bossfight	0.319048	-0.945238	-0.757143	-0.878571	-0.355714	0.385714
caveflyer	-0.033058	-0.727273	-0.685950	-0.669421	-0.221488	0.123967
chaser	0.148368	-0.890208	-0.848994	-0.919057	-0.348961	-0.049456
climber	0.631579	-0.596491	-1.000000	-0.859649	-0.263158	1.894737
coinrun	0.070671	-0.742049	-0.597173	-0.597173	0.286926	0.466431
dodgeball	-0.132653	-0.244898	0.010204	-0.102041	-0.112245	-0.030612
fruitbot	-0.373832	-0.747664	-0.780374	-0.747664	-0.262617	0.238318
heist	-0.637681	-0.420290	-0.362319	-0.275362	0.034783	-0.159420
jumper	0.196078	-0.549020	-0.196078	-0.333333	0.035294	0.862745
leaper	-0.285714	0.074286	0.457143	0.497143	0.374857	-0.091429
maze	-0.368421	-0.254386	-0.245614	-0.228070	0.021053	0.122807
miner	-0.060606	-0.454546	-0.393940	-0.424243	0.127273	0.121212
ninja	-0.097015	-0.868159	-0.644279	-0.666667	-0.153731	0.044776
plunder	0.156863	-0.768627	-0.784314	-0.784314	-0.324706	-0.078431
starpilot	-0.022989	-0.772988	-0.750000	-0.767241	-0.220690	0.178161

Table 2: Average performance for 16 games of the Procgen benchmark [Cobbe et al., 2020] across 5 random seeds. All scores are standardized by the performance of CQL [Kumar et al., 2020] on the downstream task (zero-shot generalization across the entire distribution of "easy" levels).

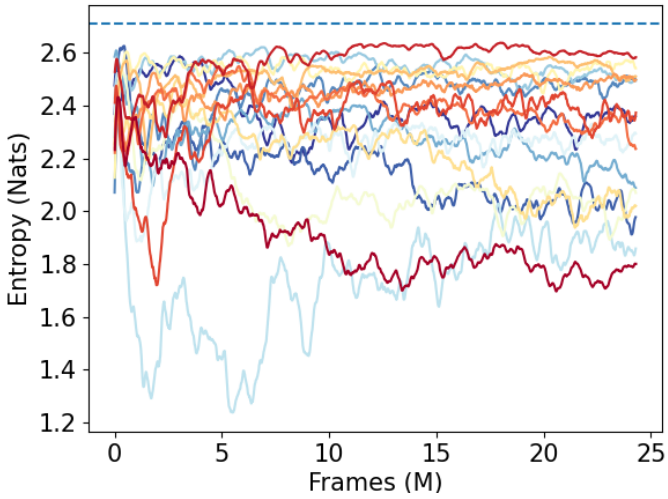

Figure 4: Average conditional entropy of online PPO policy during training phase for all 16 games of Procgen. Dotted line indicates theoretical maximum ($-\log 15$).

Choice of contrastive objective Here, we ablate the choice of the contrastive loss function used in GSF.

We define the vector distance to be the negative cosine similarity:

$$d(\boldsymbol{s}_1, \boldsymbol{s}_2) = -\frac{\boldsymbol{s}_1^\top \boldsymbol{s}_2}{||\boldsymbol{s}_1||_2 ||\boldsymbol{s}_2||_2} \tag{19}$$

The optimization objective is then taken to be the set-valued InfoNCE loss, defined for a set of positive samples $\mathcal{S}_P$, set of negative samples $\mathcal{S}_N$ and temperature parameter τ:

$$\ell_{\text{InfoNCE}}(\mathcal{S}_P, \mathcal{S}_N) = -\log \frac{\sum_{\boldsymbol{s}_1, \boldsymbol{s}_2 \in \mathcal{S}_P} \exp\left(-d(\boldsymbol{s}_1, \boldsymbol{s}_2)\tau^{-1}\right)}{\sum_{\boldsymbol{s}_1 \in \mathcal{S}_P, \boldsymbol{s}_2 \in \mathcal{S}_N} \exp(-d(\boldsymbol{s}_1, \boldsymbol{s}_2)\tau^{-1})} \tag{20}$$

The batch version of the loss is defined as

$$\ell_{\text{InfoNCE}}(\mathcal{B}) = \mathbb{E}_{\mathcal{S}_P, \mathcal{S}_N \sim \mathcal{B}}[\ell_{InfoNCE}(\mathcal{S}_P, \mathcal{S}_N)] \tag{21}$$

Fig. 5 compares the loss used by GSF with the alternative loss based on multi-class NCE. The loss based on categorical cross-entropy yields a more stable and lower error, hence we take $\ell_{\text{NCE}} = \ell_{\text{CCE}}$ in all experiments.

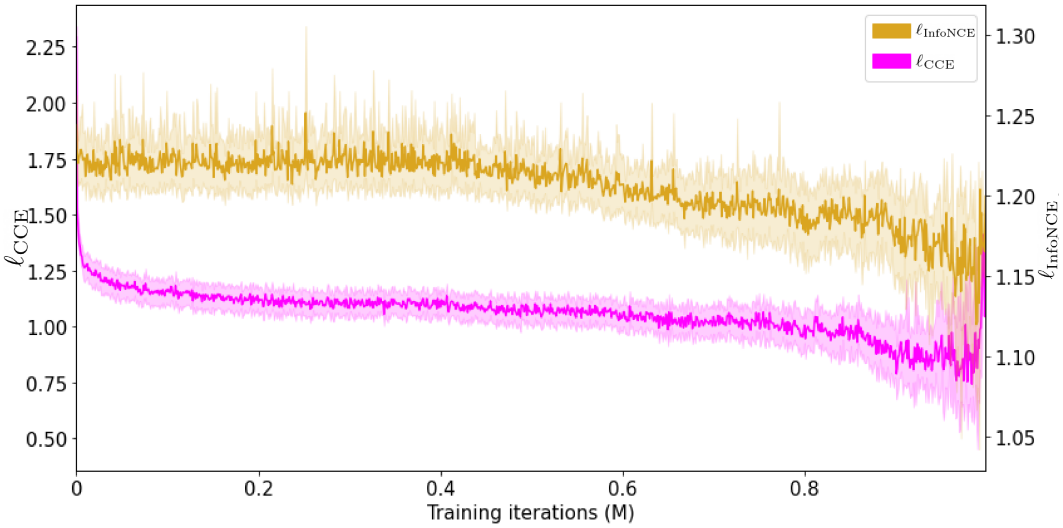

Figure 5: Comparison of two versions of our multi-class contrastive objective: (1) classification of labels via categorical cross entropy [He et al., 2020] and (2) pairwise InfoNCE [Oord et al., 2018]. Curves are averaged over 16 games and 5 random seeds.

Ablation on hyperparameters Fig. 6 shows performance of GSF for various combinations of hyperparameters, for the GVF being the Q-value of each POMDP. We picked the last combination of hyperparameters, as it has one of the highest inter-game median values, and lowest inter-quartile range (i.e. more stable performance across seeds).

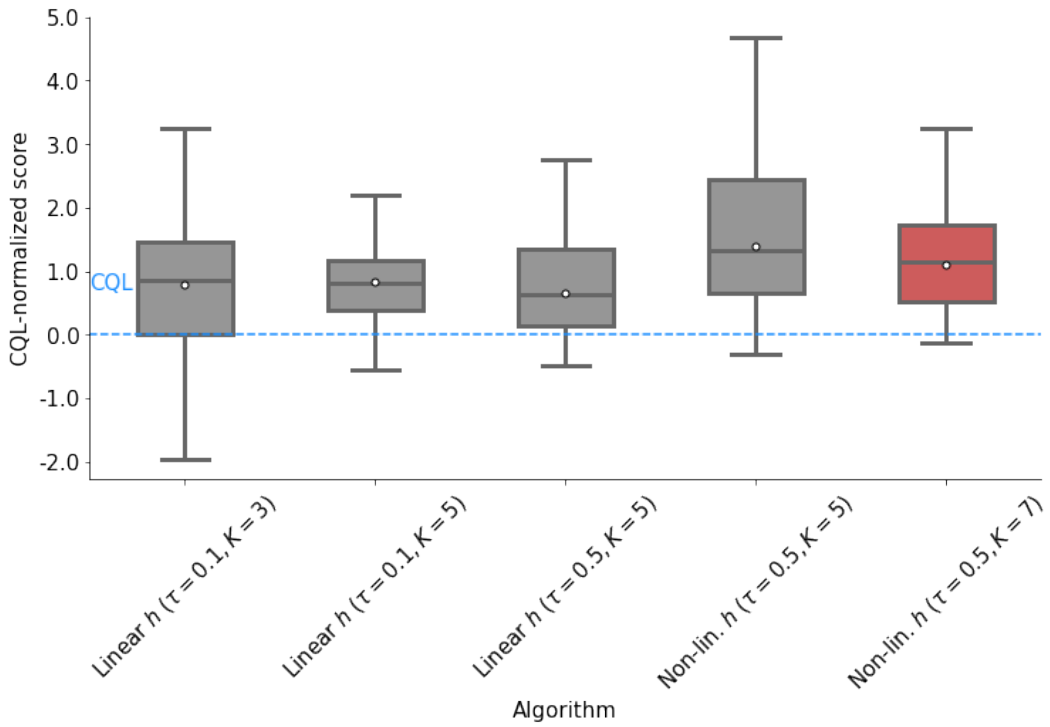

Figure 6: Ablation on GSF hyperparameters: 1) softmax temperature τ, 2) number of bins K and 3) structure of projection network h.